A VLSI TACTILE SENSING ARRAY COMPUTER[1]

John E. Tanner

California Institute of Technology

Marc H. Raibert[2] and Raymond Eskenazi

Jet Propulsion Laboratory

ABSTRACT

Here we describe a device that is at once a special purpose parallel computer and a high resolution tactile array sensor. We are interested in extending the technology that gives a robot manipulation system information about contact between the manipulator hand and its environment. We have replaced the passive substrates of earlier tactile sensors with a custom designed LSI device that handles transduction, computing, and communication. Large metal electrodes on the surface of the device are placed in contact with a conductive rubber. Deformations of this elastic material are sensed by measuring changes in its local resistivity. The sensory architecture eases the problem of connecting the transducer and computer by using an array of processors to filter and reduce raw data before communication. Since large arrays covering an entire intact wafer are planned, the design includes backup redundancy for the computing elements, and mechanisms for automatic replacement of failed elements.

---

1. This paper presents the results of one phase of research performed at the Jet Propulsion Laboratory, California Institute of Technology, sponsored by the National Aeronautics and Space Administration under Contract NAS7-100.

2. Current address is Department of Computer Science, Carnegie-Mellon University, Pittsburgh, Pa. 15213.

INTRODUCTION

Versatile computer controlled manipulation depends heavily upon the availability of precise data generated from interactions between the manipulator and its environment, and upon computational elements that convert these data into useful control information. The sense of touch is a potential source of such data, but existing tactile sensing technology falls far short of our requirements. Furthermore, techniques for mating transducer with computer -- techniques that are essential to the ultimate success of man made sensory devices -- have matured quite slowly.

Low resolution tactile sensors have been made using an array of surface electrodes on a passive substrate covered with a pressure sensitive elastic material [Bejczy 78, Briot 79]. Large, dense arrays were not feasible with those methods, not because small electrodes were hard to make, but because a separate wire connected each electrode to the sensor electronics. The package of sensor electronics itself tended to be bulky, making it hard to use near the business end of a manipulator. Furthermore, the electronics only transformed and multiplexed raw data, leaving filtering, interpretation, and recognition to be accomplished elsewhere.

Here we describe a novel design that uses custom microelectronics to overcome these problems. We are constructing a high resolution tactile sensing array by designing a special VLSI circuit that performs three basic functions:

Transduction - The exposed surface of the circuit contains sets of electrode pads, much like bonding pads, that make direct contact with a layer of pressure sensitive elastic material. The surface nature of this interaction is key to the integration of sensor and computer. It attempts to overcome the pin limitation problem that usually limits parallel computing in integrated circuits by using a two-dimensional array of inputs.

Computation - Circuits in the device form an array of computational elements, one associated with each set of surface electrodes. Each element performs simple arithmetic operations and local communication functions with neighbors, while together they form a versatile parallel 'image' processor that performs discrete two-dimensional (2D) convolutions.

Communication - Use of a distributed shift register allows all outputs from the device to be communicated over a single wire. Repeated operation of the shift register causes the entire state of the sensor to be transmitted over this compact channel.

Our strategy in using tactile array data is to treat them as images, and to apply the processing techniques that have proven useful in dealing with other types of images, especially visual images. A number of such techniques developed by workers in computer vision rely on computing the 2D convolution between an image and sets of pre-specified feature masks. Such a convolution is given by:

$$C(x,y) = \sum_{i=1}^{N} \sum_{j=1}^{N} P(x+i-1, y+j-1)\, M(i,j)$$

where $P(x,y)$ is the value of pressure on a cell at (x,y), and $M(i,j)$ is the corresponding value in an $N \times N$ filtering mask. To perform this calculation it is necessary to index the transduced data, find products, and sum the results. Such convolutions with appropriate masks permit identification and location of lines, edges, and other contours important to economical descriptions of the objects found in an image [Davis 75].

An important feature of convolution is that it can be efficiently implemented by an array of processors, each performing the same calculation as every other processor in the array, all operating simultaneously, and each only requiring data from its own local neighborhood of the image. Therefore we have implemented an array processor that performs 2D convolution between programmable masks and tactile images.

WHAT WE DID

Figure 1 illustrates the physical layout of the tactile sensor. An array of electrodes is covered by a sheet of pressure sensitive conductive rubber. This elastic material has the property that deformation causes its sheet resistivity to change in a predictable fashion. By passing a small test current from a pair of surface electrodes through the material, the magnitude of surface pressure can be measured locally.

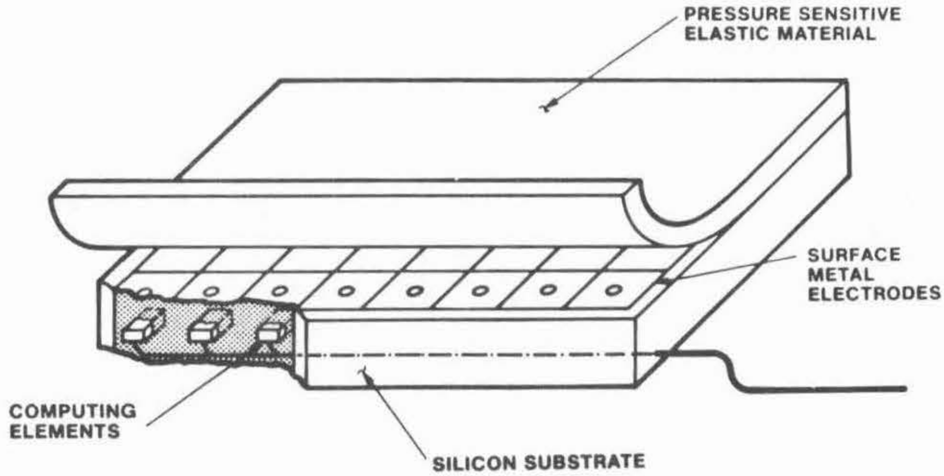

Figure 1. Mechanical architecture of touch sensor. A layer of pressure sensitive rubber is placed in contact with VLSI wafer. The surface of the wafer is covered with large sensing electrodes that are connected to circuitry that makes up an array processor within the wafer. Communication with the device takes place over a few serial lines.

In previous implementations the electrodes shown in Figure 1 were fabricated on a passive epoxy or ceramic substrate used only for mechanical support [Bejczy 78]. In our design, however, a surface layer of metal is placed on a silicon wafer to form these electrodes, while conventional nMOS circuitry below a protective glass layer forms the computing elements needed to implement transduction, filtering, data reduction, and communication. We have created a sandwich of conductive rubber, metal, and processed silicon. This is shown in Figure 2.

Figure 3 shows the overall architecture of the tactile sensing computer. An array of cells that transduce and compute are each connected to their nearest neighbors and to a global control bus. This global bus is driven externally to provide power, clock, instructions, and voltage reference to the elements of the array. The tactile cells either sense pressure, compute or communicate, with the whole array performing in lockstep under external control.

The computing units are simple but powerful processors that participate cooperatively to implement the algorithms required for tactile image sensing and analysis. Figure 4 shows a block diagram of a computing unit. Each unit contains its own set of transduction

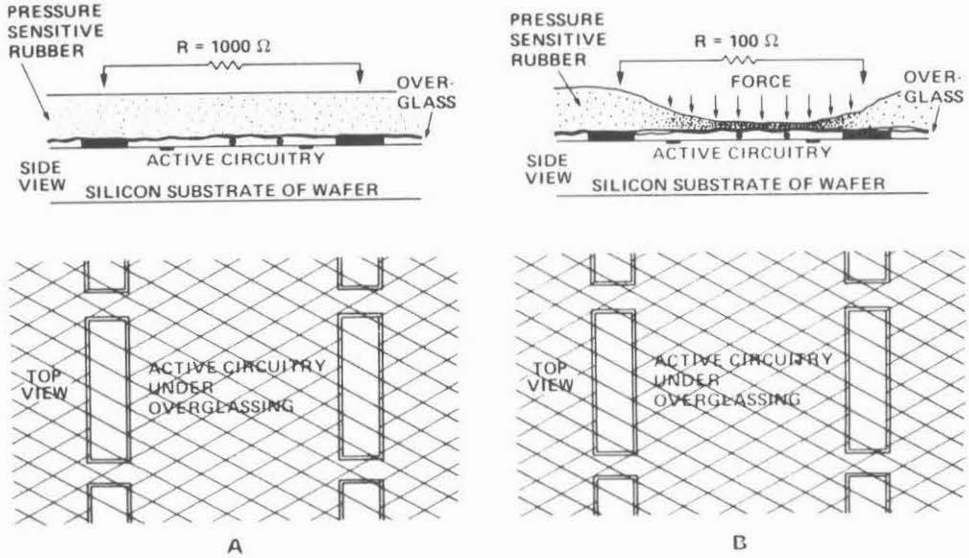

Figure 2. A) The sheet resistivity of the conductive elastic material is measured locally by metal electrodes that make contact through cuts in the overglass. B) When the material is compressed, carbon particles become more densely packed, decreasing resistivity.

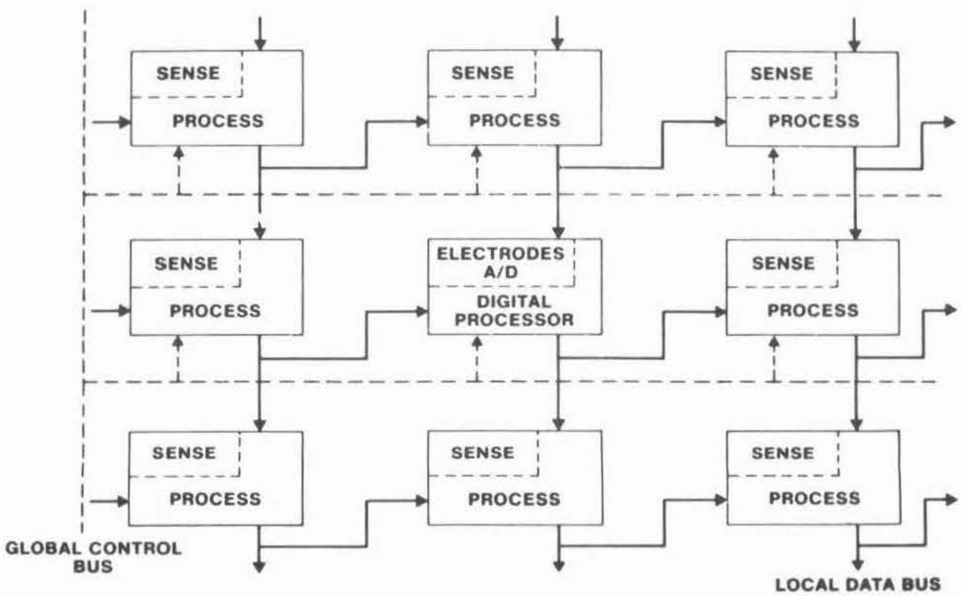

Figure 3. Block diagram of array processor. Each tactile cell has a sensing part and a computing part. A global control bus provides all cells with power, synchronization signals, and instructions. Cells are locally connected to nearest neighbors.

electrodes, an analog to digital converter, a latch, a simple arithmetic logic unit and an instruction register. The analog part converts the variable resistance of the conductive rubber into a 1-bit digital value that corresponds to the pressure on the cell. An adjustable global reference voltage allows the threshold of this analog-to-digital conversion to be varied. The result is stored locally in a latch, and can, under external program control, be made available to any of the four nearest neighbors. The latched data can also be multiplied by a number obtained from the global control bus, with the result accumulated in a 6-bit register using two's complement arithmetic. The contents of this 6-bit accumulator can be shifted and rotated in various ways.

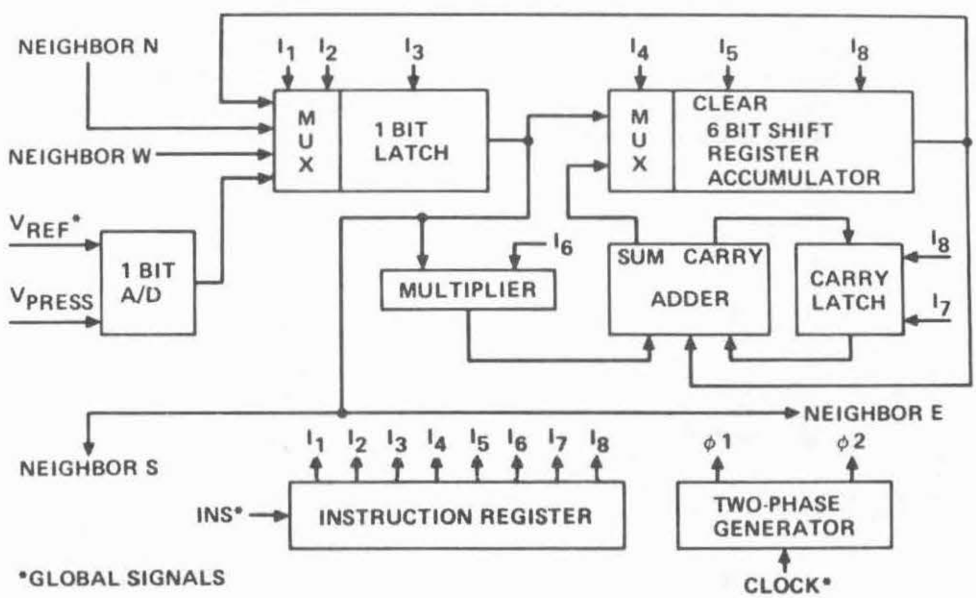

Figure 4. Block diagram of computing element.

The instruction register controls all operations of the computing element. The eight bits of the instruction are transmitted serially over a single global instruction line to the instruction register of each cell in the array. Eight instruction lines, one for each bit in the instruction register, control the various parts of the computing unit. Appendix A lists the functions of the instruction lines and gives the mnemonics and coding for the instruction set.

The computing resources just described allow the cell to sample the local pressure, to store the value, to pass data to neighboring cells, and to perform computations on the data that implement a 2D convolution with programmable mask. A simple slip detection algorithm is also easily implemented using these computational abilities. Example programs using the instruction set of Appendix A to do convolution and slip detection are given in Appendix B.

EXTENSIONS TO LARGE ARRAYS

The ultimate target of this project is to construct a 30 by 60 mm. array of 1 mm^2 tactile elements. This involves a very large area of active silicon compared to conventional designs, and therefore a much larger risk of fabricating defective circuitry. Normally, integrated circuits are manufactured by building an array of identical circuits on a silicon wafer which is subsequently diced into "chips" that are packaged and used separately. Since each chip is used separately, defective chips can be discarded on an individual basis in a straightforward manner. The overall yield can be kept high despite the presence of defective circuitry on the wafer.

Using today's fabrication techniques, a fair number of defects must be expected on a device that is 1800 mm^2. The present scheme, however, requires that all tactile sensing elements function correctly for the complete tactile sensor to operate effectively. Therefore, measures must be taken to eliminate the effects of these defects. Our designs provide a spare computing element for each tactile sensing element -- all sensing and computing circuits are replicated within the tactile cell (Figure 5). A selector circuit chooses between the pair of redundant computing elements.

We have designed a very simple 12 transistor selector that replaces a failed computing element with its backup spare. It is vitally important that this selector element be simple, since no backup is provided for the selector itself. Making it simple reduces its area and, therefore, the probability of it having a fabrication defect.

The selector is just a latch that changes state whenever its two inputs, both signals from the primary computing element, are not the same during the selector's strobe pulse. One input is the transducer output. The other is the computing element's output line. All sections of the primary computing element can be checked for proper operation by manipulating the analog reference voltage and performing appropriate computations. If a failure is detected, the secondary computing element is selected. In this design, selection takes place simultaneously and automatically for every tactile cell when the test program is executed by the array. No human intervention or hand testing is required.

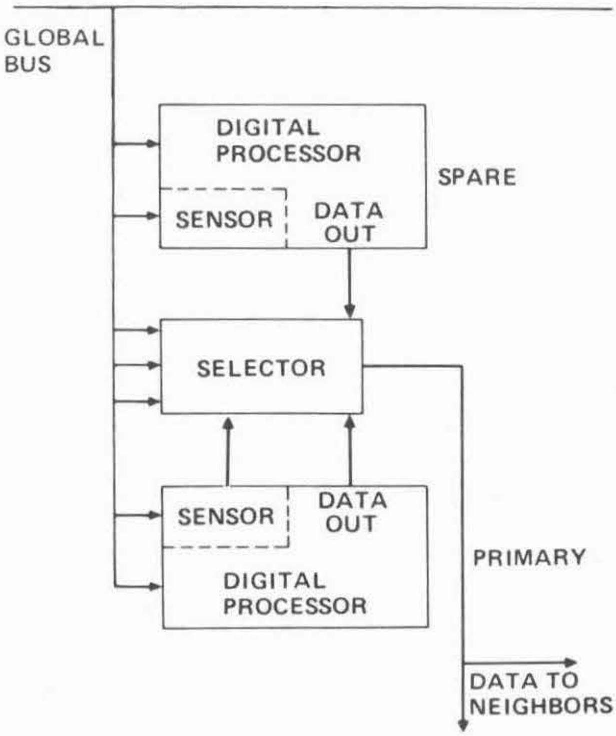

Figure 5. Block diagram of tactile cell with redundancy. Two computing elements are found within each cell. A selector enables the backup element when the primary fails a programmed function test.

Using a Poisson model of defect distribution we can estimate the yield of the array sensor. This model assumes the equal probability of a defect at any point. For a circuit with no redundancy, a defect anywhere within its area will cause it to fail. The probability P that a tactile computing element works is then:

$$P = e^{-AD}$$

where A is the area of the circuit and D is the defect density of the fabrication process.

For an array with N tactile elements the array yield will be:

$$\text{Yield} = P^N = e^{-ADN}$$

If each tactile cell contains duplicate computing elements and a selector the probability P_R that any one tactile cell is good becomes:

$$P_R = P_S - P_S(1-P)^2 + (1-P_S)P$$

where P_S is the probability a selector is good. Here we have assumed that a failed selector chooses one or the other of the computing elements it controls, rather than choosing neither or both. Also, the reliability of the global data bus is not considered. The yield of an array of N tactile cells, each with one backup computing element is then:

$$\text{Yield} = P_R^{\,N}$$

Figure 6 plots expected yield as a function of array size with defect density of 0.05/mm^2, selector area of 0.045 mm^2, and computing element area of 0.90 mm^2 for both redundant and non-redundant cells. It can be seen from these plots that arrays containing 1000 elements are achievable.

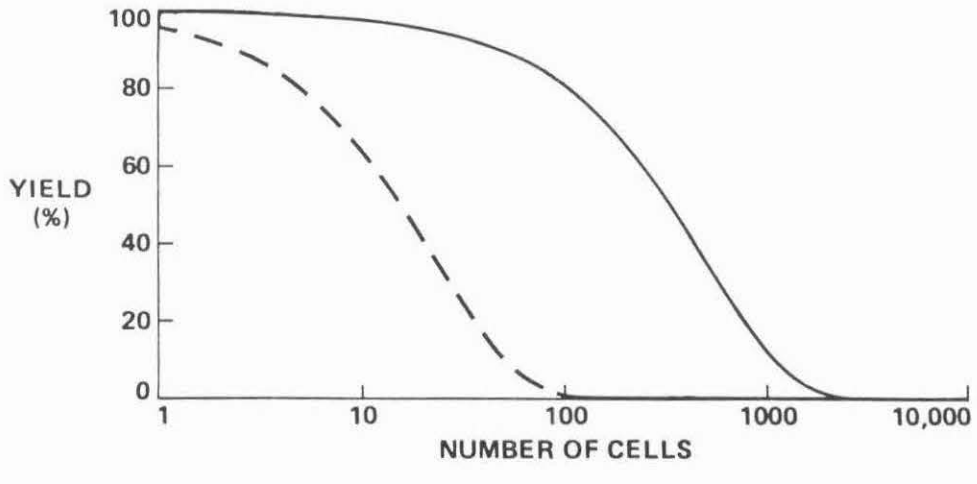

Figure 6. The expected yield of a tactile sensing array is calculated using a Poisson model of defect distribution. It is plotted here as a function of N, the number of cells in the array. The dotted curve is the yield with no redundancy. The solid curve is the yield with one spare computing element per cell as described in text. Defect density is 0.05/mm^2, selector area is 0.045mm^2 and computing element area is 0.9mm^2.

Figure 7 is a floorplan and photograph of a chip that implements a 1x2 tactile sensing array computer. In order to see the active circuitry the pressure sensitive material is not present in this photograph. The chip includes two tactile cells, each of which has two computing elements and a selector. Bonding wires are attached to conventional input/output pads along the top and bottom. These are used to test this early prototype. The large metal areas in the center of each tactile cell are the sensing electrodes. The size of this chip is about 2mm x 3mm.

This first prototype was fabricated as part of MPC5-80, the Multi-Project Chip effort coordinated by the Xerox Palo Alto Research Center [Conway 81]. A 3x3 array was fabricated as part of MOSIS, a similar service coordinated by ISI [Cohen 81]. Both versions are nMOS with lambda equal to 2.5 micron. Table 1 gives the measured performance of the 3x3 tactile sensing array.

Pressure Sensitivity	50 grams
Clock Rate	3 MHz
Instruction Time	3 μsec
6 Bit Add (Multiply)	18 μsec
3x2 Convolution	140 μsec
Power Requirements	5 V
	10 mA/Cell

TABLE 1. PERFORMANCE OF TACTILE SENSING CHIP

PROBLEMS AND PLANS

An important problem with our present nMOS implementation is the excessive power requirement for circuits with large active areas. For small arrays, the current of 10 mA/cell is reasonable, but an 1800 element array, our original target, would require 18 amps. We are designing a new version of the tactile sensor in CMOS. This fabrication technology is expected to become available to us soon through the ISI implementation system. A CMOS design should greatly relieve the power supply and dissipation problems inherent in nMOS.

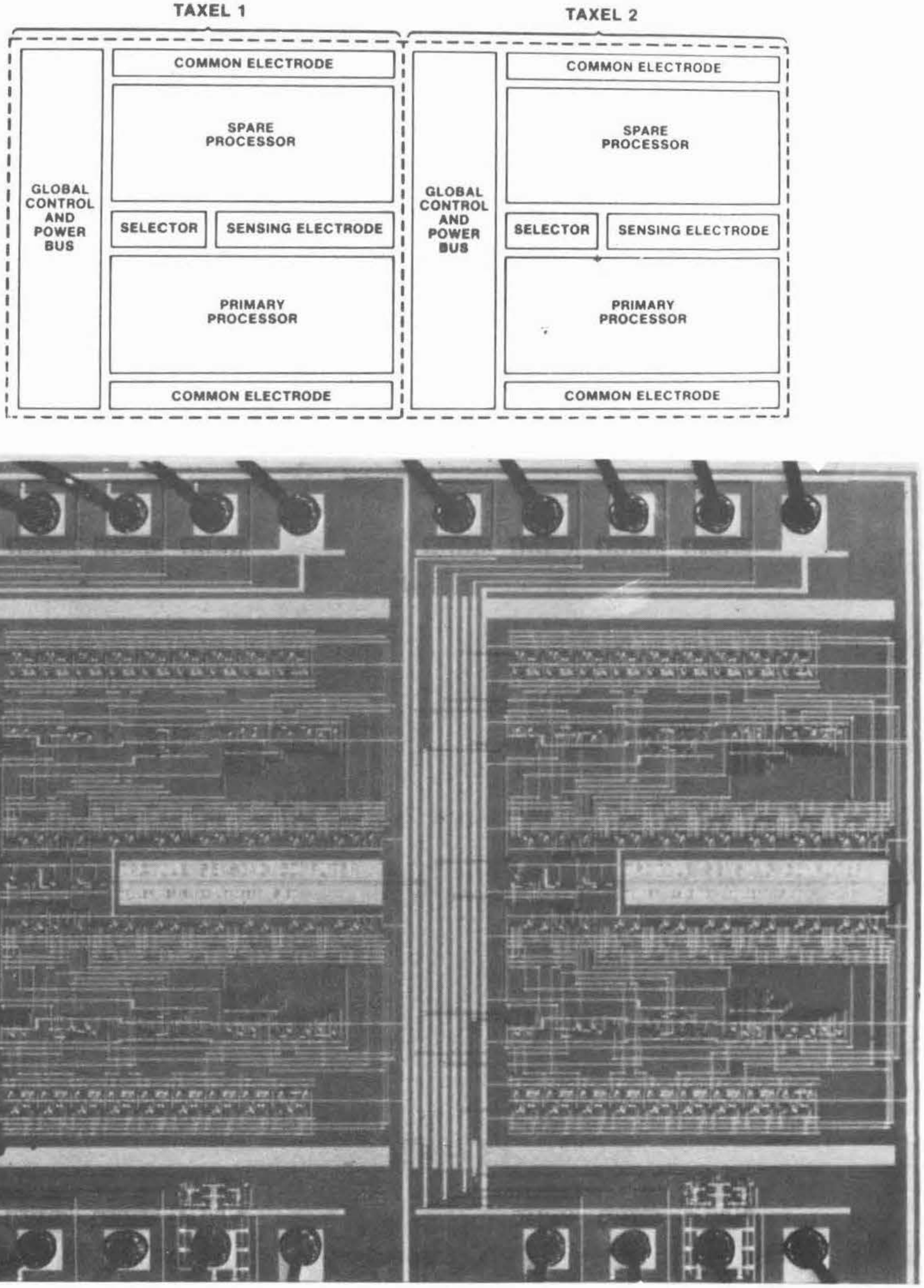

Figure 7. Floorplan and photograph of a tactile sensing chip produced by MPC5-80.

Several variations on the basic transduction scheme are being considered for future implementation. One of these uses a layer of the fabrication process as a piezoresistive transducer. This layer could be a custom layer added after the circuit has been fabricated conventionally. It may be possible to utilize the polysilicon layer as the pressure sensitive element, since it typically exhibits a slight piezoresistive effect. This technique is used in some pressure transducers available commercially. Another possible method of detecting pressure is to use a layer of material that changes its optical properties with pressure. The circuitry below must then detect the optical difference. The advantage of this technique is that it allows a greater isolation of the integrated circuitry from the physical and chemical dangers of the environment.

We also hope to develop other methods of fault tolerance. The information content of an image is degraded only slightly by the loss of one pixel. In the tactile sensing array if one cell fails, all the data from other cells that are to be shifted through the bad cell are lost. This results in an image with a stripe defect. We are developing a more complex shifting pattern that routes each cell's information toward the edge of the array through multiple pathways. This new design requires the ALU within each cell to reconstruct correct values periodically in the shifting pattern by computing the majority function of the values that arrive over different paths. Simulations show that this method of redundant communication gives an array a 40 fold increase in immunity to stripe defects.

Combining transduction with electronic circuitry in an intimate way gives rise to a set of problems unique to this application of VLSI. The transducer must be in contact with the environment it is sensing, in this case, the gripping surface of a robot's hand. The electronics, which in most applications are protected from the environment by a sturdy sealed package, is now an integral part of the transducer. The silicon substrate must be able to withstand any force that the robot's hand is expected to encounter and to resist any chemical contamination diffusing through the pressure sensitive elastic covering. In addition, heat sinking must be provided to the substrate to remove the heat produced by such a large area of active circuitry. Further testing will show how critical these packaging problems are to producing a practical, useful tactile sensor.

So far we have not thoroughly characterized the sensor's analog response to forces and distortions. This must be done. The several different types of available pressure sensitive elastic materials will be tested with a special electrode chip -- one that has six different electrode geometries. Sensitivity, range, and localization are the variables of interest.

SUMMARY

Combining transduction with computation in the same device is an effective way to use the power of VLSI technology, and to overcome the traditional sensing problems of interconnection, communication and computation. The two-dimensional nature of the integrated circuit nicely matches the surface nature of the tactile sensing problem and the architecture of array processors. The convolution algorithm used to refine raw image data exploits the full concurrency of an array processor to achieve high performance. Wafer scale integration is necessary to build working arrays of useful size, and fault tolerant techniques are necessary to achieve wafer scale integration. These ideas have been brought together to start a new generation of tactile sensors for robots.

ACKNOWLEDGEMENTS

The authors wish to thank Glen Okita and Dean Uehara for many early contributions to this work.

REFERENCES

Bejczy, A. K., "Smart Sensors for Smart Hands," AIAA/NASA Conference on "Smart Sensors," November 1978, Hampton, VA.

Briot, M., "The Utilization of an Artificial Skin Sensor for the Identification of Solid Objects," 9th International Symposium on Industrial Robots, Washington, D.C., March 1979.

Cohen, D., "MOSIS -- The ARPA Silicon Broker," Proceedings of the Caltech Conference on VLSI, January 1981.

Conway, L. A., "The MPC Adventures: Experiences with VLSI Implementation Systems," Proceedings of the Caltech Conference on VLSI, January 1981.

Davis, L. S., "A Survey of Edge Detection Techniques," Computer Graphics and Image Processing, 1975, 4, pp. 248-270.

Appendix A

The eight bits of the instruction word control different parts of the cell and can be identified as follows:

- I_1 Latch input select 0
- I_2 Latch input select 1
- I_3 Latch shift/store
- I_4 Accumulator MSB select
- I_5 Accumulator clear
- I_6 Multiplier value
- I_7 Carry Clear
- I_8 Accumulator and Carry latch shift/store

Listed below are some useful instructions and their mnemonics.

	I_1 I_2 I_3 I_4 I_5 I_6 I_7 I_8	(x = don't care)
CLEAR	x x 0 x 1 x 1 0	Clear accumulator and carry latch
CLRA	x x 0 x 1 x 0 0	Clear accumulator only
CLRC	x x 0 x 0 x 1 0	Clear carry latch only
SHFTS	0 1 1 x 0 x 0 0	Shift all latch values one cell south
SHFTW	1 0 1 x 0 x 0 0	Shift all latch values one cell west
STADC	1 1 1 x 0 x 0 0	Store the value from the a/d converter in the latch
ADDM1	x x 0 0 0 1 0 1	Multiply latch value by 1 and add to accumulator LSB
ADDMO	x x 0 0 0 0 0 1	Multiply latch value by 0 and add to accumulator LSB
ROTAT	0 0 1 1 0 x 0 1	Rotate the accumulator through the latch
LTCHA	x x 0 1 0 x 0 1	Move the value in the latch into the accumulator MSB
NOP	x x 0 x 0 x 0 0	No operation

To multiply the value in the latch by a mask value 5 and add the result to the six bit accumulator requires this sequence.

```
ADDM1
ADDMO
ADDM1        5 = 000101(binary)
ADDMO
ADDMO
ADDMO
```

Appendix B

One possible mask for detecting vertical edges is shown in the example below. The pressure data show that an object with a rectangular corner is pressing on the lower left part of a 6 x 6 array of cells. The results of the convolution are 36 values, one computed by each cell, as shown on the right. The non-zero values indicate a vertical edge.

Pressure						Mask			After Filtering					
0	0	0	0	0	0				0	0	0	0	0	0
0	0	0	0	0	0				0	0	0	0	0	0
0	0	0	0	0	0	5	-5		0	0	0	0	0	0
1	1	1	0	0	0	5	-5	=>	0	0	5	0	0	0
1	1	1	0	0	0	5	-5		0	0	10	0	0	0
1	1	1	0	0	0				0	0	15	0	0	0
1	1	1	0	0	0				0	0	15	0	0	0

Figure B1. Example of edge detection using convolution.

Listed below is a program written using the mnemonics of Appendix A to perform the edge detection convolution of Figure B1.

INSTRUCTION	COMMENTS
CLEAR	NOTE: We are cell (1,1)
STADC	Get pressure data P(1,1)
ADDM1	M(1,1) = 5
ADDMO	
ADDM1	
ADDMO	
ADDMO	
ADDMO	
CLRC	
SHFTW	Get neighbor's data P(2,1)
ADDM1	M(2,1) = -5
ADDM1	
ADDMO	
ADDM1	
ADDM1	
ADDM1	
CLRC	
SHFTS	Get neighbor's data P(2,2)
ADDM1	M(2,2) = -5
ADDM1	
ADDMO	

. . . and so on for a total of 48 instructions

Figure B2. Program to perform 3x2 convolution.

Another computation that may prove useful is the determination of change in pressure from one moment to the next. This can indicate slip or be one step in a data compression scheme. Each cell can detect a change of pressure by computing the inequality function:

$$S = \begin{cases} 0 , & \text{for } P(T_1) = P(T_2) \\ 1 , & \text{for } P(T_1) \neq P(T_2) \end{cases}$$

where $P(T_1)$ is the pressure on the cell at time T_1 and $P(T_2)$ is the pressure on the cell at a later time T_2. For single bit pressure values, the function S can be computed by adding $P(T_1)$ to $P(T_2)$. The least significant bit of the result is S. The program listed in Figure B3 implements this algorithm.

INSTRUCTION	COMMENTS
CLEAR	
STADC	Get pressure data
ADDM1	Put it in accumulator LSB
ADDMO	(Multiply by 1)
ADDMO	
ADDMO	
ADDMO	
ADDMO	
STADC	Get new pressure data
ADDM1	Add it to accumulator
ADDMO	
ADDMO	
ADDMO	
ADDMO	
ADDMO	
ROTAT	LSB is 1 if slip occurred between samples

Figure B3. Program to calculate slip.